Amendments to the Claims:

This listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims:

1.-82. (cancelled)

83. (new) A method for creating a global simulation model of an architecture, comprising:

reading an architecture description file of a global model and storing, in a component and connection rule table, in a connection coherency role table, and in a source file formatting table, information related to all of the possible configurations of said architecture, said information comprising a plurality of components, said plurality of components comprising software simulation elements being chosen by a user from a library of various component types and in a library of environment components, to create the global simulation model of the architecture corresponding to a functional specification defined in a configuration definition file, wherein said plurality of components conform to a specification of the architecture of the global model specified by an architecture description file containing the information related to all possible configurations, with each said component being assigned a name that unambiguously identifies its position in the architecture, and each said component further comprising a type selected from among the following: Active Components, Monitoring and Verification Blocks, Intermediate Blocks, System Blocks; and Global Blocks;

instantiating the components specified in the configuration definition file by a user-developer using a list of the components, designated by their names and types and including parameters or executing procedures, the configuration definition file comprising a file from which components are selected together with their types and optional additional indications concerning the type of interface and the server involved in the configuration to be generated by a Configurator, and storing the components specified in the configuration definition file in an instance connection table;

topologically connecting the instances and storing the corresponding information in the instance connection table;

generating wiring interface signals, at the level of each instance of the components, by applying regular expressions, stored in the component and connection rule table, based on names of signals contained in a wiring table; and

generating, using the instance connection table, the wiring table, and the formatting table, HDL-type and HLL-type source files of the global simulation model corresponding to a configuration specified by the configuration definition file;

wherein the method is accomplished using a data processing system configured in accordance with the Configurator, and wherein the global simulation model of the architecture comprises models of integrated circuits under development for the realization of a machine, and models that create a test and verification environment defined in the configuration definition file of the components of the architecture.

84. (new) A method according to claim 83, further comprising transmitting to the HLL-type source file a portion of each said component including:

the name (LABEL) of the component;

the type of the instance (DUT, XACTOR, VERIFIER, MONITOR); and

an HDL path comprising a hierarchical name of the component in the description of the model.

85. (new) A method according to claim 83, wherein the configuration definition file also includes a keyword identifying a server by name or number in which one of said components is instantiated when the method is used in a multi-server system.

86. (new) A method according to claim 84, wherein the configuration definition file also includes a keyword identifying a server by name or number in which one of said components is instantiated when the method is used in a multi-server system.

87. (new) A method according to claim 86 further comprising:

dividing a configuration into several HDL-type and HLL-type components;

sorting the divided HDL-type components and HLL-type components according to the servers to which they belong;

generating peripheral components of HDL-type used for sending and receiving signals between the parts of the configuration;

duplicating the Global Blocks using the Configurator and the instantiation of the Global Blocks duplicated in each server; and

generating HLL-type components that serve as a communication medium between the servers.

88. (new) A method according to claim 85, wherein the automatic connection between components comprises:

selecting one or more of said components and their respective positions; and

creating the actual connection between said selected components,

generating a wiring table that associates the signals connected to one another with a unique name of a wire that connects them; and

generating source files for the HDL-type and HLL-type components to complete a wiring phase.

89. (new) A method according to claim 88, further comprising:

a. connecting the Global Blocks and the System Blocks to all of the selected components;

b. connecting the signals between the selected components;

c. making an additional pass to connect the remaining unconnected signals of each selected component to one or more predetermined signals in order to produce a given stable state; and

d. generating, using the Configurator, one or more partial configurations each of which comprises a subset of the architecture.

90. (new) A method according to claim 89, wherein the predetermined signals are signals of the System Block corresponding to one of the selected components.

91. (new) A method according to claim 83, wherein the architecture description file of the global model includes one or more simulation models of Global Blocks and System Blocks, said Global Blocks and System Blocks being connected to one another and adapted for handling environment signals.

92. (new) A method according to claim 91, wherein the System Blocks are connected to other components and supply said other components with system signals that are specific to said other components.

93. (new) A method according to claim 92, further comprising performing a conformity check of the connections, and comparing the connection table of the real instances between blocks to the connection coherency rule table.

94. (new) A method according to claim 93, further comprising comparing the physical connections between the components to the connection coherency rule table, in order to detect any incompatible connections between the ends of the connections between the components, and in cases where an incompatible connection is detected, specifying and adding an adapter component (Intermediate Block) to the instance connection table, said adapter component being inserted into the incompatible connection between the components.

95. (new) A method according to claim 94, wherein the configuration definition file includes information, specified by an attribute, concerning the utilization of adapter components (Intermediate Blocks) with the instances of active Components whose connections are compared to the instance connection table, in order to detect any incompatible connections between the ends of the connections between the components and, in cases where an incompatible connection is detected,

specifying and adding an adapter component (Intermediate Block) to the instance connection table, said adapter component being inserted into the incompatible connection between the components.

96. (new) A method according to claim 95, further comprising selecting certain connections between the components of the connection coherency rule table, specifying additional connections constituting branches leading to respective additional models which represent tools (probes) for monitoring the connections, and adding said additional connections to the instance connection table.

97. (new) A method according claim 83, wherein the Configurator is configured to include a source file generation phase in which the HDL-type and HLL-type source files are generated based on the content of the component and connection role table, the coherency rule table, the source file formatting table, the instance connection table, and the wiring table.

98. (new) A method according to claim 97, wherein the data processing system is configured to execute an operation using the Configurator for each of a plurality of configuration variants, in order to obtain a corresponding plurality of simulation models, each said configuration variant and its corresponding simulation model corresponding to the same functional specification written in a description comprising one or more combinations of languages of different levels (HDL, HLL).

99. (new) A method according to claim 83, wherein the data processing system is configured to generate a functional specification of the global simulation model in a computer format compatible with a high-level programming language, and in a format compatible with a hardware description language.

100. (new) A method according to claim 99, wherein the configuration definition file comprises, for each said component, at least one part in HDL-type language, said part in HDL-type language providing an interface with other models.

101. (new) A method according to claim 100, wherein the models that include said at least one part in HLL-type language include interface adapters.

102. (new) A method according to claim 101, wherein the Configurator is configured to select each interface adapter model as a function of the connection coherency rule table.

103. (new) A method according to claim 102, wherein the connections of the physical signals are specified by one or more Ports, each port being an arbitrary selection of the signals of the HDL-type interface of one of said components by means of regular expressions based on the names of said physical signals, and each port including regular expression/substitute expression pairs, said regular expressions being successively applied to the name of each signal of the HDL-type interface, and if the final substitution is identical for two signals, the signals are connected to one another, the connection being stored in the wiring table.

104. (new) A method according to claim 103, wherein each said interface adapter is shared among several models connected to the same port, and only one of said models transmits signals through said port.

105. (new) A data processing system for automatically generating a global simulation model of a configuration of software simulation elements, comprising:

a plurality of said software simulation elements mutually connected by inter working connections so as to constitute the global simulation model of an architecture,

wherein the global simulation model comprises models of integrated circuits for the realization of a machine that conforms to functional specification of a configuration defined in a configuration definition file by a user,

a Configurator further comprising

means for creating a simulation of wiring by executing stored regular expressions, and means for using the configuration definition file, a component and connection rule table, a connection coherency rule table, the component and connection rule table being written in a high level language, and the component and connection rule table describing properties of software components for simulating the circuit, the connection coherency rule table being written in a high level language, and

means for instantiating components resulting from the configuration definition file, and an HLL code generator that combines the parameters of the components with the connection roles.

106. (new) A system according to claim 105, wherein the components comprise Active Components, Monitoring and Verification Blocks, Intermediate Blocks, System Blocks, and Global Blocks.

107. (new) A system according to claim 106, further comprising means to perform a conformity check of the connections by comparing the instance connection

table with a table of coherency rules for the physical connections between the models chosen from the blocks constituting the global model.

108. (new) A system according to claim 107, further comprising means to compare the instance connection table to the connection coherency rule table to detect any incompatible connections between the ends of the connections between blocks, and incases where an incompatible connection is detected, the data processing system is configured to specify and add an adapter component (Intermediate Block) to the instance connection table, said adapter component being inserted into the incompatible connection between the components.

109. (new) A system according to claim 108, wherein the component and connection rule table includes the properties of the components and contains global parameters common to all of the component types and exists in the form of a table distributed into one or more associative tables, wherein the entries are names designating all of the possible models for the same component.

110. (new) A system according to claim 109, wherein the associative tables are adapted to contain a description either in the form of parameter sets or in the form of references to procedures that generate the required values, wherein the entries of these associative tables comprise names designating all of the possible models for the same component, and form a character string containing predetermined special identifiers that are replaced by values calculated by the Configurator.

111. (new) A system according to claim 110, wherein at least three selectors indicate the instance to be used, and in which the following selectors are transmitted as parameters to a constructor of an HLL object:

a first selector indicating a current instance (item);

a second selector specifying the current instance connected to the other end of the port; and

a third selector indicating a composite instance corresponding to an active Component containing an observation port.

112. (new) A system according to claim 105, wherein the Configurator comprises one or more connection coherency rule tables representing the rules for interconnecting the components and for inserting intermediate components; one or more component and connection rule tables representing the system-level connection rules and the rules for generating connections between the signals; and one or more source file formatting tables representing the rules for generating instances of HLL-type objects.

113. (new) A system according to claim 105, wherein the Configurator comprises:

an HLL base class uniquely identifying each object instantiated and polling the configuration;

means for generating and automatically instantiating System Blocks;

means for using tables to associate the signals connected together under a unique name of the connecting wires; and

means for using a formatting table to generate HDL-type and HLL-type source files.

114. (new) A system according to claim 105, wherein the system is configured to receive from an operator a functional specification of the configuration

in a high level language, and to complete the functional specification with the components in language having level lower than said high level language.

115. (new) A system according to claim 105, wherein the following entries in a hash define a Component Type and correlate each Component Type to a hash, said hash comprising the following:

a first entry comprising a name of the HDL module of the component and a name of a corresponding source file; and

a second entry comprising a definition of a method for selecting the signals that are part of a Port, said definition comprising a set of entries indexed by the name of the Port;

wherein the Configurator is configured to associate each said Port name with a table of regular expressions and a pointer to a signal connection procedure that controls the application of the expressions to the names of the signals of the interface of the component.

116. (new) A system according to claim 115, wherein said Component Type comprises one or more Active Components having a generic structure that includes a Block containing the HDL description and a Block in HLL that provides the access paths to the HDL resources, and a description of the block in HLL;

wherein the set of signals of the HDL block constitute the interface of the containing Block, formed by Ports, which are arbitrary logical selections of the signals of an interface, and also formed by interface adapters which are the software parts that handle, in each Port, the two-way communication between the parts in HLL and those in HDL, the interface adapters being selected by the Configurator.

117. (new) A system according to claim 116, wherein the Ports are specified in the form of regular expressions that select subsets of signals to be connected and define connection rules.

118. (new) A system according to claim 105, wherein the Configurator is further configured to generate Transfer Components which are inserted on each side of the interface between servers, said Transfer Components comprising wires for inputs and registers for outputs.

119. (new) A method for automatically generating a global simulation model of an architecture comprising models of integrated circuits under development, comprising:

reading an architecture description file of the global simulation model containing information related to all possible configurations of said architecture;

storing said information related to all possible configurations;

instantiating one or more components and storing corresponding information in an instance connection table;

topologically connecting a plurality of interface signals,

wiring the plurality of interface signals at the level of each instance of the one or more components using a component and connection rule table;

storing information corresponding said wiring in a wiring table; and

generating one or more HDL-type source file and one or more HLL-type source file of the global simulation model, each said HDL-type source file and each said HLL-type source file corresponding to the configuration specified by the configuration definition file.

120. (new) A method according to claim 119, further comprising transmitting to the HLL-type source file parts of each component including:

a name of the component;

a type of the instance; and

an HDL path comprising a hierarchical name of the component in the description of the model.

121. (new) A method according to claim 120, wherein the configuration definition file also includes a. keyword indicating the name or number of the server in which one of said components is instantiated when the method is used in a multi-server system.

122. (new) A method according to claim 121, wherein the configuration definition file also includes a keyword indicating the name or number of the server in which one of said components is instantiated when the method is used in a multi-server system.

123. (new) A method according to claim 122, further comprising:

dividing a Configuration into a plurality of HDL-type components and a plurality of HLL-type objects;

sorting the plurality of HDL-type components and the plurality of HLL objects according to the servers to which they belong;

generating one or more HDL-type peripheral components used for sending and receiving signals between the parts of the configuration; and

duplicating the Global Blocks, the instantiation of the Global Blocks being duplicated in each server; and

generating the plurality of HLL objects to serve as a communication medium between the servers.

124. (new) A method according to claim 119, wherein the architecture description file of the global model includes simulation models of Global Blocks and System Blocks, said Global Blocks and System Blocks being connected to one another and adapted for handling environment signals.

125. (new) A method according to claim 124, wherein the System Blocks are connected to other components and supply said other components with system signals that are specific to said other components.

126. (new) A method according to claim. 125, wherein the data processing system is configured to perform a conformity check of the connections by comparing the connection table of real instances between blocks to the connection coherency rule table.

127. (new) A method according claim 126, wherein the Configurator system generates the source files in HDL language and in HLL language, in a source me generation phase, based on the content of the component and connection rule table, the coherency rule table, the source file formatting table, the instance connection table and the wiring table.

128. (new) A method according to claim 127, wherein the data processing system is configured to generate, using the Configurator system, for each of a

plurality of configuration variants, a plurality of simulation models, each of said plurality of simulation models corresponding to a same functional specification, but written in a description comprising one or more combinations of languages of different levels.

129. (new) A method according to claim 128, wherein the data processing system is configured to generate a functional specification of the global simulation model in a computer format that is compatible with a high-level programming language and a format compatible with a hardware description language.